Available online at www.sciencedirect.com

ScienceDirect

Theoretical Computer Science 362 (2006) 196–206

Theoretical Computer Science

www.elsevier.com/locate/tcs

Deterministic M2M multicast in radio networks☆

Leszek Gąsieniec[a,*], Evangelos Kranakis[b], Andrzej Pelc[c], Qin Xin[d]

[a] *Department of Computer Science, The University of Liverpool, Liverpool, L69 7ZF, UK*
[b] *School of Computer Science, Carleton University, Ottawa, Ont., Canada K1S 5B6,*
[c] *Dép. d'informatique, Université du Québec en Outaouais, Gatineau, Qué., Canada J8X 3X7*
[d] *Department of Informatics, The University of Bergen, P.B. 7800, N-5020 Bergen, Norway*

Received 7 March 2005; received in revised form 2 June 2006; accepted 14 June 2006

Communicated by M. Jerrum

Abstract

We study the problem of exchanging messages within a fixed group of k nodes, called *participants*, in an n-node radio network, modeled as an undirected graph. This communication task was previously considered in the setting of ATM video applications, in [J.M. Tsai, H.-H. Fang, C.-Y. Lee, A multicast solution for ATM video applications, IEEE Trans. Circuits Systems Video Technol. 7 (1997) 675–686], and was called multipoint-to-multipoint (M2M) multicasting. While the radio network topology is known to all nodes, it is assumed that no node is aware of the location of the participants. We give a distributed deterministic algorithm for the M2M multicasting problem in radio networks, and analyze its time complexity. We show that if the maximum distance between any two out of k participants is d then this local information exchange problem can be solved in time $O(d \log^2 n + k \log^4 n)$.

Keywords: Radio communication; Deterministic algorithm; Broadcasting; Multicasting

1. Introduction

Wireless networks are expected to support many group communication applications (such as distance learning, video conferencing, disaster recovery and distributed collaborative computing). In such applications, any subset of nodes, called *participants* may be required to send messages to one another. The task of exchanging messages within a fixed group of nodes in a network is called *Multipoint-to-Multipoint* (*M2M*) *multicasting* (cf. [32]). Efficient execution of this task in radio networks is the main topic of this paper.

1.1. Model and terminology

A radio network is a collection of stations, equipped with capabilities of transmitting and receiving messages. The network is modeled as an n-node undirected connected graph $G = (V, E)$, where the set of nodes V represents stations.

☆ A preliminary version of this paper was presented at ICALP 2004.

* Corresponding author. Tel.: +44 151 7947910; fax: +44 151 7943715.
E-mail addresses: leszek@csc.liv.ac.uk (L. Gąsieniec), kranakis@scs.carleton.ca (E. Kranakis), pelc@uqo.ca (A. Pelc), qinxin@csc.liv.ac.uk (Q. Xin).

0304-3975/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2006.06.017

Each node has a unique label from the set $[N] = \{0, 1, \ldots, N-1\}$ of integers, where N is bounded by some polynomial in n. An edge e joins nodes u and v, if and only if, the transmitter of u can reach v and vice versa. Nodes send messages in synchronous *steps*. In every step every node acts either as a *transmitter* or as a *receiver*. A node acting as a transmitter sends a message to all of its neighbors in the graph G. A node acting as a receiver in a given step gets a message, if and only if, exactly one of its neighbors transmits in this step. If at least two neighbors v and v' of u transmit simultaneously in a given step, none of the messages is received by u in this step. In this case we say that a *collision* occurred at u. It is assumed that the effect at node u of more than one of its neighbors transmitting in a given step is the same as that of no neighbor transmitting, i.e., a node cannot distinguish a collision from silence.

Broadcasting and *gossiping* are two classical problems of information dissemination in computer networks. In broadcasting, we want to distribute a message from a distinguished *source* node to all other nodes in the network. In gossiping, each node v in the network initially holds a message m_v (input value of v), and we wish to distribute all messages m_v to all nodes in the network. M2M multicasting is a natural generalization of gossiping, in which information exchange does not concern all nodes of the network but only a subset of all nodes, i.e., participants. Messages circulating in the network are arbitrarily long finite binary strings. In particular, a node can send any concatenation of input values m_v, possibly equipped with appropriate routing information, as one message, in one step. One of the main efficiency criteria of an algorithm achieving any communication task (such as broadcasting, gossiping, or M2M multicasting), is the *time complexity* of the algorithm, defined as its number of steps performed until the termination condition is satisfied (see Section 1.2).

In this paper we consider deterministic communication algorithms that use the entire knowledge about the network topology. Such algorithms are useful in radio networks that have a reasonably stable graph of connections. As long as no changes occur in the network topology during the execution of the algorithm, the communication task can be accomplished successfully.

1.2. The problem

The precise definition of the task of M2M multicasting in a radio network is the following. A set P of k nodes, called participants, is specified. Each node knows whether it belongs to P, knows the size of P, but does not know P itself. Initially, each participant v has a message m_v (input value), which has to be delivered to all other participants. Nodes outside of P play only the role of relayers. M2M multicasting terminates when each participant gets input values of the $k-1$ other participants.

Although either broadcasting or gossiping could be used to solve M2M multicasting, the use of these procedures is likely to be inefficient because an application may involve only a small set of participants, compared to the total number of nodes in the underlying radio network. In this paper we address the problem of minimizing the time complexity of an algorithm achieving M2M multicast in radio networks. To the best of our knowledge, this is the first study of M2M multicast time in this communication model.

1.3. Previous work

Most of the work devoted to communication algorithms in radio networks concerns the tasks of broadcasting and gossiping. In the model where the network topology is known to all nodes, Gaber and Mansour [19] showed that the broadcasting task can be completed in time $O(D + \log^5 n)$, where D is the diameter of the network. Two alternative broadcasting algorithms (superior for small diameters) with the running times $O(D \log^2 n)$ and $O(D \log n + \log^2 n)$ can be found in [6,28] respectively. Elkin and Kortsarz [16] showed efficiently computable radio broadcast schedules that work in time $D + O(\log^4 n)$ in general graphs and in time $D + O(\log^3 n)$ in planar graphs. These bounds were lately improved by Gąsieniec et al. in [21] to $D + O(\log^3 n)$ and $3D$, respectively. Moreover, a randomized broadcast procedure with the expected running time $D + O(\log^2 n)$ can also be found in [21]. The computation of an optimal radio broadcast schedule for an arbitrary network is known to be NP-hard, even if the underlying graph of connections is embedded in a plane [5,30].

Many authors [4,7,8,10,11,13,26,12] studied deterministic distributed broadcasting in radio networks, in which every node knows only its own label, in the model of directed graphs. Increasingly faster broadcasting algorithms working on arbitrary n-node (directed) radio networks were successively constructed, with the currently fastest being the $O(n \log^2 D)$-time algorithm from [12]. (Here D is the radius of the network, i.e., the longest distance from the

source to any other node). On the other hand, in [11], a lower bound $\Omega(n \log D)$ on broadcasting time was proved for directed n-node networks of radius D.

The gossiping problem was not studied in the context of radio networks of known topology, until relatively recent work of Gąsieniec and Potapov [22]. They studied the gossiping problem in radio networks of known topology, where each message is limited to a bounded number of bits. In this model several time-efficient gossiping algorithms were proposed in various standard network topologies, including lines, rings, stars and trees. It was also proved in [22] that there exists a radio network topology in which gossiping (with bounded messages) requires $\Omega(n \log n)$ time. Very recently, Gąsieniec et al. [23] studied gossiping in radio networks with known topology and arbitrarily large messages, and several optimal gossiping algorithms were proposed for a wide range of radio topologies.

So far, the gossiping problem was mostly studied in the context of radio networks, where the topology of connections is unknown to nodes. In this model, Chrobak et al. [10] proposed a deterministic algorithm that completes the gossiping task in time $O(n^{3/2} \log^3 n)$. For small values of the diameter D, the gossiping time was later improved by Gąsieniec and Lingas [20] to $O(nD^{1/2} \log^3 n)$. An $O(n^{3/2})$-time gossiping algorithm (an improved version of the gossiping algorithm from [10]) can be found in [33]. A very recent $O(n^{4/3} \log^3 n)$-time gossiping algorithm was proposed by Gąsieniec et al. in [24]. A study of deterministic gossiping in radio networks with unknown topology and messages of bounded size, can be found in [9]. Randomized algorithms for the gossiping problem in radio networks of unknown topology also attracted attention of many authors. In [10], Chrobak et al. proposed a gossiping algorithm with $O(n \log^4 n)$ expected time. Even faster randomized algorithms were later proposed, with expected times $O(n \log^3 n)$ in [29], and $O(n \log^2 n)$ in [12].

1.4. Our results

Our main result is the design of an efficient algorithm for the M2M multicasting problem in radio networks, and the analysis of its time complexity. We study this task for k participants in an n-node radio network. We show that if the maximum distance between any two out of k participants is d then this problem can be solved in time $O(d \log^2 n + k \log^4 n)$ by a deterministic algorithm. Note that the time complexity of our algorithm is affected by the size n of the network only in a rather weak way (by factors polylogarithmic in n). The only linear factors in the formula are those concerning the set of participants: their number k and maximum distance d between them. Our solution is based on a novel application of the graph clustering method preserving locality [19] and on efficient adaptive collision resolution based on the concept of promoters, see Section 2.2.

2. Paradigms and tools

In this section we first present a high-level idea of the algorithms (both for trees and for arbitrary graphs), and then we describe combinatorial tools used to implement their main stages.

2.1. Overview of the algorithms

In the first part of the algorithms (in both cases), the input values are gathered in one selected *meeting point*. The input values traveling towards the meeting point, from time to time compete with other input values for the same communication channel. We will guarantee *the invariant* that each input value competes with any other input value at most once. In the second part of the multicast procedure, a compound message containing all input values is distributed to all participants.

Although the algorithms used for trees and for arbitrary graphs share the same general structure, they significantly differ in details of their design. The two main differences lie in the choice of the meeting point and in the way in which the competition for the same communication channel is resolved.

In trees, the selection of the meeting point is implicit. Before the communication process is started, one node is chosen as the root of the tree. During the M2M multicast, all input values of participants move towards this root. The meeting point is the first node which gathers all input values. In fact, the meeting point is the lowest common ancestor (LCA) of all participants, with respect to the chosen root of the tree. Note that the distance between the LCA and all participants is always limited to d. Each competition is resolved with the help of a system of synchronized descending selectors (see Sections 2.2 and 2.3).

In arbitrary graphs, the choice (computation) of the meeting point is much more complex. Not knowing the position of participants, we cannot fix the meeting point in advance, since—in the worst case—messages would have to travel along the diameter of the entire network before meeting each other. Instead, we propose a new clustering concept, that allows us to group all participants in one of the clusters with a relatively small diameter, comparable with d. Each cluster has its own meeting point and a BFS spanning tree rooted in it. In each cluster, similarly as in the case of trees, we try to move all messages from the participants towards the meeting point. However, efficient traversal limited to branches of the BFS tree is not always possible. This is due to the fact that in the cluster there exist edges outside of the BFS tree that potentially cause a lot of collisions. Thus the competition is becoming much harder. In order to overcome this problem, we propose a special algorithm that resolves collisions between competing messages. This algorithm is based on a novel use of descending selectors, combined with broadcasting and gossiping procedures.

2.2. Resolving competition

The main difficulty in radio communication is the presence of collisions. It has been shown, see e.g., [11,10], that efficient tools for collision resolution can be designed on the basis of combinatorial structures possessing a *selectivity property*. We say that a set R hits a set Z at element z, if $R \cap Z = \{z\}$, and a family of sets $\mathcal{F}$ hits a set Z at element z, if $R \cap Z = \{z\}$ for at least one $R \in \mathcal{F}$. In [11] the authors used a family of subsets of set $\{0, 1, \ldots, N-1\} = [N]$ which hits each subset of $[N]$ of a size at most $k \leqslant N$ at all of its elements. They refer to this family as *strongly k-selective* family. It is known that strongly k-selective families coincide with the notion of $(k-1)$-*cover free families* [17], *disjunctive codes* [15], and *superimposed codes* [25]. E.g., in [18] it is proved that the minimum size of a k-selective family is $\mathrm{O}(k^2 \log N)$. In [10] the authors define a family of subsets of the set $[N]$ which hits each subset of $[N]$ of a size at most k on at least $k/2$ distinct elements, where $N \geqslant k \geqslant 1$. They call it a *k-selector* and prove (using the probabilistic method) the existence of such a family of size $\mathrm{O}(k \log N) = \mathrm{O}(k \log n)$. An alternative construction and a tighter analysis of the size of k-selectors can be found in the context of the combinatorial group testing problem in [14].

In what follows we show how to cope with collisions occurring during the competition process. This will be done with the help of selective families and selectors. A communication mechanism based on a family S of sets of nodes consists in using elements of the family S as sets of transmitters in consecutive steps of the algorithm.

2.2.1. Promoting messages in unknown stars

Assume that l nodes from $V' = \{v_1, v_2, \ldots, v_l\}$ are immediate neighbors (not aware of each other) of another node w, i.e., they form a star with a center in w, and they all compete (at some stage of the algorithm) to move their message to w. The process of moving messages from nodes in V' to w is called a *promotion*. It is known that the mechanism based on l-selectors allows at least half of the nodes in V' to deliver their messages to w in time $\mathrm{O}(l \log n)$ [10]. Indeed, when a set in the selector hits a set of competing nodes at a node z, this node delivers its message.

Let $S(l)$ represent the collision resolution mechanism based on l-selectors. Note that $S(l)$, if applied in undirected networks, can be supported by the *acknowledgment of delivery* mechanism in which each transmission from the neighbors of w is alternated with an acknowledgment message coming from the central node w. If during the execution of $S(l)$ a transmission towards w is successful, i.e., one of $v_i \in V'$ succeeds in delivering its message, the acknowledgment issued by w and returned to all nodes in V' contains the label of the successful node; otherwise the acknowledgment is null. Note that acknowledgments do not interfere with competitors' transmissions because separate steps of the procedure are devoted to them. Let $\mathbf{S}(l)$ be the mechanism with the acknowledgment feature based on $S(l)$. In other words, the use of $\mathbf{S}(l)$ allows us to exclude from further transmissions all nodes in V' that have managed to deliver their message to w during the execution of $\mathbf{S}(l)$. Note that the duration of $\mathbf{S}(l)$ is $\mathrm{O}(l \log n)$.

Let $S^*(i)$ be the communication mechanism based on concatenation of i selectors $S(2^i), S(2^{i-1}), \ldots, S(2^1)$. We will call it a *descending selector*. The descending selector extended by the acknowledgment mechanism, i.e., the concatenation of $\mathbf{S}(2^i), \mathbf{S}(2^{i-1}), \ldots, \mathbf{S}(2^1)$, forms a communication procedure called *a promoter*, denoted by $\mathbf{S}^*(i)$. Note that the duration of $\mathbf{S}^*(i)$ is $\mathrm{O}(2^i \log n)$.

Lemma 1. *If $V' = \{v_1, v_2, \ldots, v_l\}$ is a set of neighbors of w, and all nodes in V' use the same promoter $\mathbf{S}^*(i)$, where $l \leqslant 2^i$, then all nodes in V' deliver their messages to w in time $\mathrm{O}(2^i \log n)$.*

Proof. The proof is done by induction, and is based on the fact that after the execution of each $\mathbf{S}(2^j)$, for $j = i, \ldots, 1$, the number of competing nodes in V' is $\leqslant 2^{j-1}$. □

2.2.2. Promoting messages in unknown bipartite graphs

Assume that we have a connected bipartite graph B, in which nodes are partitioned into two sets U and L. In our further considerations, sets U and L will correspond to two adjacent BFS levels, upper and lower respectively, in a subgraph of G. All nodes know the topology of B. For any node $x \in L$ there is exactly one node $y \in U$ called the *parent* of x. Conversely, every node in U is a parent of some node in L. This relationship is also known to all nodes in B. Note that since nodes in U can be parents of several nodes in L, we have $|U| \leqslant |L|$. Let l be an upper bound on $|L|$.

We now specify the procedure ENHANCED-PROMOTION(l) to be described below. The procedure works for known l. Prior to its execution, some nodes in L are *active*, i.e., they have messages to be transmitted to U. Let L' denote the set of active nodes in L, and let $U' \subseteq U$ denote the set of their parents. Upon the completion of the procedure, all messages from nodes in L' are gathered in a single node r' in U'. When this happens, node r' becomes active and all nodes in L' become non-active.

Procedure ENHANCED-PROMOTION(l);
1. All nodes in L' communicate with their parents using the promoter $\mathbf{S}^*(i)$, for $i - 1 < \log l \leqslant i$.
2. All nodes in $L' \cup U'$ take part in leader election choosing the node r' with the smallest label in U'. This is done using procedure FINDMAX from [10].
3. Node r' performs broadcasting to all other nodes in $L' \cup U'$. This is done using algorithm DOBROADCAST from [10]. The broadcasting tree T rooted at r' is created, in which children are aware of their parents in T.
4. Each node (except the node r') communicates with its parent in T, using the promoter $\mathbf{S}^*(i)$, for $i - 1 < \log l \leqslant i$. After this step, all nodes are aware of their children in T.
5. The node r' sends a token traversing all edges of T by Depth-First Search. The token collects all messages from L', places them in r' and informs all nodes in $L' \cup U'$.

Lemma 2. *The procedure* ENHANCED-PROMOTION(l) *gathers all messages from L' in r' and works in time* $\mathrm{O}(l \log^3 n)$.

Proof. Step 1 is based on a single use of the promoter $\mathbf{S}^*(i)$, for $i - 1 < \log l \leqslant i$. By Lemma 1, all nodes from L' successfully communicate with their parents in time $\mathrm{O}(2^i \log n) = \mathrm{O}(l \log n)$.

Step 2 is a direct application of procedure FINDMAX from [10]. This procedure performs leader election (finding the node with smallest label) in l-node radio networks with unknown topology, where labels are from the set $[N]$. The procedure works in time $\mathrm{O}(l \log^3 N) = \mathrm{O}(l \log^3 n)$.

Step 3 is a direct application of algorithm DOBROADCAST from [10]. This algorithm performs broadcasting in l-node radio networks with unknown topology, where labels are from the set $[N]$ in time $\mathrm{O}(l \log^2 N) = \mathrm{O}(l \log^2 n)$.

Step 4 is analogous to Step 1. It is performed in time $\mathrm{O}(l \log n)$.

In step 5 the DFS traversal of the tree T is done in time $\mathrm{O}(l)$. Upon its completion all messages of nodes from L' are gathered in r'. The total time of procedure ENHANCED-PROMOTION(l) is $\mathrm{O}(l \log^3 n)$. □

2.3. The time multiplexing mechanism

Both the promotion and the enhanced promotion procedures work under the assumption that an upper bound l on the number of competing nodes is known, and that all nodes start executing the procedure simultaneously. However, in our applications these assumptions may not be valid. In order to overcome these obstacles, we use the time multiplexing of procedures. Suppose that procedures $P_1, \ldots, P_s$ with running times $t_1, \ldots, t_s$ are given, where $t_{i+1} = 2t_i$, for $i < s$. The time *multiplexer* of these procedures is a new procedure M that permits to run them "concurrently" and periodically as follows. The jth step of the multiplexer M is the $(\lfloor j/s \rfloor \bmod t_{j \bmod s})$th step of the procedure $P_{j \bmod s}$. Thus, the execution of two consecutive steps in the same procedure P_i is interleaved with the execution of single steps of every other procedure P_j, $j \neq i$. Moreover, the execution of different procedures is synchronized, i.e., during a single execution of the procedure P_i, exactly two executions of the procedure P_{i-1} are performed.

In our applications we will use multiplexers of promotion and enhanced promotion procedures to gather all input values in the meeting point. The number s of procedures will be $\mathrm{O}(\log N) = \mathrm{O}(\log n)$. The following two lemmas are

the counterparts of Lemmas 1 and 2 in the case when the number of competing nodes l is unknown and the starting times of transmissions for these nodes are arbitrary.

In the case of promotion, procedure P_i is $\mathbf{S}^*(i)$, possibly padded with some empty steps at the end. This is to ensure the fixed duration of each P_i, equal to $t_i = \mathrm{O}(2^i \log n)$.

Lemma 3. *Let $V' = \{v_1, v_2, \ldots, v_l\}$ be a set of neighbors of w of unknown size l. Assume that at some arbitrary time step t_0, each node of the set V' has a message. Then, using the multiplexer of promoters $\mathbf{S}^*(i)$, where $1 \leqslant i \leqslant \lceil \log N \rceil$, all nodes in V' deliver their messages to w by the time step $t_0 + \mathrm{O}(l \log^2 n)$.*

Proof. After receiving an input value of a participant, a node v waits until the first step of $\mathbf{S}^*(1)$ (the first promotion procedure within the multiplexer), and then executes consecutive steps of $\mathbf{S}^*(1)$, omitting other steps of the multiplexer. If procedure $\mathbf{S}^*(1)$ succeeds to achieve promotion, node v stops using the multiplexer. Otherwise, i.e., when the number of competing nodes is too large for $\mathbf{S}^*(1)$, node v waits until the first step of $\mathbf{S}^*(2)$ within the multiplexer, executes only steps of $\mathbf{S}^*(2)$, and so on, until successful with some $\mathbf{S}^*(i)$. This mechanism guarantees a successful promotion without knowing the upper bound l on the number of competing nodes. Since $\mathbf{S}^*(i)$ is the first successful promoter, we know that $2^{i-1} < l \leqslant 2^i$. Thus the time of promotion (including the waiting periods and unsuccessful transmission attempts with $\mathbf{S}^*(j)$, for $j < i$) is linear in $t_i \cdot \mathrm{O}(\log n) = \mathrm{O}(l \log^2 n)$. The factor $\mathrm{O}(\log n)$ comes from the fact that $\mathrm{O}(\log n)$ promoters are interleaved in the multiplexer. □

In the case of enhanced promotion, multiplexing of procedures ENHANCED-PROMOTION(2^i) is not enough to guarantee the invariant that each input value competes with any other input value at most once. We need to ensure that all messages from nodes in L' are delivered to a single node r', and delivery acknowledgment is received by all nodes in L'. It is possible that the set L' is partitioned into subsets $L_1, \ldots, L_m$ which are not aware of each other upon completion of the procedure ENHANCED-PROMOTION(2^i), due to collisions in transmissions. This could happen if the procedure ENHANCED-PROMOTION(2^i) is applied on a set L' of size larger than 2^i. In order to maintain the invariant we need to hold the promotion process until the call of the procedure ENHANCED-PROMOTION($2^{\lceil \log |L'| \rceil}$) is executed.

This problem is solved by adding an additional testing procedure Q_i immediately after the procedure ENHANCED-PROMOTION(2^i). Let $B_1, B_2, \ldots, B_m$ be connected subgraphs of B, such that B_j is induced by nodes in L_j and their parents in U'. Upon completion of the procedure ENHANCED-PROMOTION(2^i), every subgraph B_j has its leader λ_j whose label will play the role of a label of the whole subgraph B_j.

Let $\mathcal{R}$ be a 2-strongly selective family $\{R_1, \ldots, R_y\}$ of size $y = \mathrm{O}(\log N)$ on the set $[N]$. Transmissions in the testing procedure will be of two types. Transmissions of type 1 are those from step 5 of the procedure ENHANCED-PROMOTION(2^i). They occupy a block of 2^i consecutive steps. Transmissions of type 2 also occupy a block of 2^i consecutive steps: in each of these steps every node performing this type of transmission sends its label. The testing procedure Q_i consists of y blocks, each of duration of 2^i steps. In the mth block, if $\lambda_j \in R_m$ then all nodes from B_j execute transmissions of type 1 (i.e., each node in B_j performs consecutive transmissions prescribed by step 5 of the procedure ENHANCED-PROMOTION(2^i)). Otherwise, i.e., if $\lambda_j \notin R_m$ then all nodes from B_j execute transmissions of type 2, i.e., they send their labels in every step of the block. Intuitively, the aim of the above testing procedure is to inform nodes in subgraphs B_j of the presence of other subgraphs $B_{j'}$ connected with B_j by at least one edge. (Recall that the entire graph B is connected). Note that if the subgraph B_j is connected by an edge to some other subgraph $B_{j'}$, there will be a block m in the application of the strongly 2-selective family when $\lambda_j \in R_m$ and $\lambda_j \notin R_m$ (and vice versa). In this case the traversal of the token in the subgraph B_j will be interrupted, which is enough to figure out that B_j does not form the whole graph of competitors.

We are now ready to present the multiplexer in the case of enhanced promotion. The multiplexed procedures P_i are defined as follows. For $i = 1, \ldots, \lceil \log N \rceil$, P_i is the procedure ENHANCED-PROMOTION(2^i), immediately followed by Q_i and possibly padded with some empty steps at the end. This is to ensure the fixed duration of each P_i, equal to $t_i \cdot \mathrm{O}(\log n) = \mathrm{O}(2^i \log^4 n)$.

Lemma 4. *Assume that at some arbitrary time step t_0, each node of the set L' has a message, where $|L'| = l$. Then, using the multiplexer of procedures P_i, where $1 \leqslant i \leqslant \lceil \log N \rceil$, all messages from L' are gathered in r' during the execution of the same P_i. This happens by the time step $t_0 + \mathrm{O}(l \log^4 n)$.*

Proof. The execution of procedure P_i, for $l \leqslant 2^i$, guarantees gathering all messages from L' in r'. Some messages from L' may get to nodes in U' before the others, during the execution of P_j, for $j < i$. However, if not all messages from L' are promoted to a single node in U' during the execution of P_j, the testing procedure Q_j does not permit promotion. Hence gathering occurs during the execution of a single procedure P_j, for some $j \leqslant i$.

Procedure ENHANCED-PROMOTION(2^i) is executed in time $O(2^i \log^3 n)$, in view of Lemma 2. The execution time of the procedure Q_i is the product of the size $O(\log n)$ of a strongly 2-selective family by the size 2^i of a block of transmissions. Hence the duration of P_i is $O(2^i \log^3 n)$. Multiplexing procedures P_i adds an extra multiplicative factor $O(\log n)$, as in Lemma 3. This concludes the proof. □

2.4. Graph clustering preserving locality

The main purpose of the clustering method is to obtain a representation of a large graph as a collection of its much smaller subgraphs (clusters), while preserving local distances between the nodes.

Let $G = (V, E)$ be a graph representing a radio network. Initially we pick an arbitrary node c in V that becomes a *central node* in G. The radius of G is the maximum distance D between c and any other node, and d is the maximum distance between the participants. The clustering method groups nodes belonging to some connected subgraph G', in the same cluster C. If the diameter of G' is d, the diameter of C is at most $O(d \log n)$.

Definition 1. Let l_j be the jth BFS level in a graph G with respect to a central node c, i.e., $l_j = \{v | dist(c, v) = j\}$.

Definition 2. For any positive integer x, we denote by $\pi(x)$ the partition of nodes of G into super-levels, such that, each super-level is composed of $4d$ consecutive BFS levels, where the first super-level starts from an arbitrary but fixed BFS level l_x (note that levels $l_0, l_1, \ldots, l_{x-1}$ are excluded from the partition $\pi(x)$). More precisely, the ith super-level in $\pi(x)$ is $G_i(x) = \{v | v \in l_j, (i-1-x) \cdot 4d \leqslant j \leqslant (i-x) \cdot 4d - 1\}$, for $i = 1, 2, \cdots, \lceil (D-x)/4d \rceil$, where D is the radius of G with respect to the central node c, and d is the maximum distance between the participants. Given a super-level $G_i(x)$, its top level is $l_{(i-1-x)\cdot 4d}$, and its bottom level is $l_{(i-x)\cdot 4d-1}$. Note that $G_i(x)$ is not necessarily connected.

Definition 3. For each node u belonging to the top level of $G_i(x)$, the pre-cluster $S_u^{(i)}$, is the set of all nodes in $G_i(x)$ at distance $\leqslant 4d$ from u.

The *clusters* are obtained by growing appropriate pre-clusters, according to the mechanism used in the *Cover Algorithm* presented in [1,19]. We say that two sets A and B of nodes are at distance δ if δ is the minimum distance between any pair of nodes a and b, such that $a \in A$ and $b \in B$. The growing algorithm is performed in $O(\log n)$ stages. In each stage $i = 1, \ldots, O(\log n)$, a collection of clusters C_*^i (at distance 2 apart) is created as follows. We start with an arbitrary pre-cluster which will be contained in the cluster C_0^i. At each step of the extension procedure we add to the cluster C_0^i a new layer of pre-clusters that intersect with C_0^i or are at distance at most 1 from C_0^i. Note that this extension is successful only if the number of new nodes coming with the new pre-clusters is at least as big as the number of nodes in the pre-clusters already present in the cluster C_0^i. If this condition is not met, the extension of the cluster C_0^i is terminated, i.e., the construction of C_0^i completes without augmenting nodes available in the just considered layer of pre-clusters. Instead, the pre-clusters in the new layer are moved for consideration in stage $i + 1$. The process of growing clusters $C_1^i, C_2^i, \ldots$ is performed similarly, and it continues as long as we have at least one pre-cluster that neither forms a part of any cluster constructed in stages $1, \ldots, i$, nor has been moved for consideration in stage $i + 1$.

Lemma 5. *The clusters have the following properties*:
1. *Each cluster is a union of some pre-clusters.*
2. *Each pre-cluster is a member of exactly one cluster.*
3. *Each cluster is a connected subgraph of G.*
4. *The diameter of each cluster is* $O(d \log n)$.
5. *There is a* $O(\log n)$*-coloring of the clusters, such that clusters having the same color are at distance* $\geqslant 2$ *apart.*

Proof. Properties 1–3 follow directly from the construction of the clusters. Property 4 is based on the fact that each pre-cluster has diameter $\leqslant 4d$. Moreover, during the construction of any cluster, the number of new layers of pre-clusters

is limited to $\lceil \log n \rceil$, since each extension by a new layer of pre-clusters at least doubles the number of nodes in the pre-clusters of the currently constructed cluster. The coloring in Property 5 is defined as follows: clusters constructed in the same round are given the same color. Since they are at distance 2 apart, and the number of rounds is bounded by $\log n$, the property is satisfied. □

Definition 4. A 2-partition of the graph G comprises two different partitions: $\pi(0)$ which starts at the super-level $G_1(0)$, and $\pi(2d)$ which starts at the super-level $G_1(2d)$.

Lemma 6. *In at least one of the partitions of the* 2*-partition*, *there exists at least one cluster that contains all* k *participants and the shortest paths between them. Moreover*, *in this partition*, *any other cluster containing some* (*or all*) *of the* k *participants*, *is colored differently* (*according to the coloring from Lemma* 5).

Proof. Let v be one of the k participants. According to our definition of the *2-partition*, we can prove that the node v must fall into the central $2d$ BFS levels of a super-level in one of the partitions, except for the case when v belongs to the first d BFS levels (when all k participants belong to the cluster based on the central node c). Thus, there exists a node p at the top level of the corresponding super-level $G_i(\cdot)$, which is at distance $dist(p, v) \leqslant 3d$ from the node v. Since all other participants are at distance $\leqslant d$ from v, there exists a pre-cluster (which constitutes a part of a cluster) $S_p^{(i)}$ which contains the entire set of k participants. The second part of the lemma follows from the fact that clusters having the same color cannot overlap. □

3. The description of the M2M multicast algorithms

We start this section with the presentation of a M2M multicasting algorithm designed for radio networks with a tree topology. M2M multicast in trees works in time $O(d + k \log^2 n)$-time. We later present a more complex M2M multicast algorithm which works in arbitrary graphs in time $O(d \log^2 n + k \log^4 n)$.

3.1. M2M multicast in trees

Our M2M multicast algorithm in trees works as follows. All participants send their input values towards a commonly selected root of the tree. The first node that gets input values from all k participants becomes the *meeting point*, which then distributes the combined message to all participants. In order to avoid collisions caused by transmissions at adjacent levels, we enforce an extra rule that nodes at level j (at distance j from the root r) execute their transmissions in steps i, where $i = j \pmod 3$. This slows down the whole process only by a multiplicative constant 3.

Algorithm TREE-MULTICAST(T)
1. All nodes agree on the root r of the tree T, which is the node with the smallest label in T. Levels in T are defined with respect to this root.
2. Input values of participants traverse, level by level, towards r. After getting a new input value, a node v first transmits it to its parent in the tree. If the acknowledgment of successful transmission does not arrive (within three steps), the node v starts using the multiplexer of promoters $\mathbf{S}^*(1), \ldots, \mathbf{S}^*(\lceil \log n \rceil)$.
3. The first node c that gathers input values of all k participants (the *meeting point*), broadcasts the compound message to all participants, along branches of T.

Theorem 1. *Algorithm* TREE-MULTICAST(T) *completes the M2M multicast in a radio network* T *with tree topology in time* $O(d + k \log^2 n)$.

Proof. Step 1 does not involve communication. Since all nodes know the topology of T (including the labels of nodes), they use the same deterministic algorithm to choose the node r with the smallest label.

In Step 2 participants' input values traverse the tree towards node r. Note that the meeting point c is the LCA of all participants, with respect to r. We show that the last input value enters this node in time $O(d + k \log^2 n)$. The node c is at distance at most d from each of the participants. Consider a single input value. When it moves towards the root, it traverses each edge with constant delay (transmitting to the parent and waiting for acknowledgment), if there is no

competition. These transmissions contribute a summand $O(d)$ to the time complexity. If at any time the input value competes with some other l messages, it is promoted to the next level of the tree in time $O(l \log^2 n)$, in view of Lemma 3. Note that two messages competing once will never compete against each other again, since later on, they travel on the same branch of the tree. This means that the total time spent by an input value on competing with other messages is bounded by $O(k \log^2 n)$. Thus the last message arrives at the meeting point in time $O(d + k \log^2 n)$. This shows that Step 2 is executed in time $O(d + k \log^2 n)$.

Step 3 is a naive broadcasting procedure that distributes the compound message to all nodes (including all participants) within distance d from the node c. Since there are no collisions in radio broadcasting in trees, the compound message is distributed to all participants in time at most d. This concludes the proof. □

3.2. M2M multicast in arbitrary graphs

In this section we show how to perform M2M multicast in arbitrary radio networks in time $O(d \log^2 n + k \log^4 n)$. The algorithm is based on the clustering method introduced in Section 2.4, on the multiplexer of enhanced promotion procedures followed by testing procedures, see Section 2.3.

In the system of clusters, there exists at least one and at most $\lceil \log n \rceil$ clusters with diameter at most $\lceil d \log n \rceil$ that contain all k participants (see Section 2.4). In what follows, we consider transmissions performed inside a single cluster. Recall that simultaneous execution of transmissions in clusters having the same color does not cause collisions between the clusters, because all clusters of the same color are at distance at least 2 apart. In order to avoid collisions between clusters with different colors, we execute transmissions for different colors in $O(\log n)$ (number of colors) different stages. This gives an $O(\log n)$ slowdown in comparison with an execution in a single cluster. Note that having the partition into clusters ready, we could now perform the M2M multicast in time $O(k \cdot d)$ polylog n, applying a leader election algorithm and broadcasting k times. However, our intention is to design a $O((k + d)$ polylog $n)$ algorithm.

The communication in a cluster C of a 2-partition is performed as follows.

Algorithm GRAPH-MULTICAST(C)

1. Select the node r in C with the smallest label and a spanning BFS tree T of C rooted at r.
2. Messages sent by the participants travel, level by level, towards the root r. After getting a new input value, a node v first transmits it to its parent in the BFS tree. If the acknowledgment of successful transmission does not arrive (within three steps), the node v starts using the multiplexer of procedures ENHANCED-PROMOTION(2^i) followed by testing procedures Q_i, for $i = 1, \ldots, \lceil \log n \rceil$.
3. The root r distributes the compound message to all participants, using the broadcasting procedure in graphs with known topology [21].

Lemma 7. *Algorithm* GRAPH-MULTICAST(C) *completes M2M multicast in any cluster C of diameter d' in time* $O(d' + k \log^3 n)$.

Proof. Step 1 does not involve communication, since the topology of G is known to every node.

In Step 2, similarly as in Algorithm TREE-MULTICAST(T), the participants' input values either traverse levels of the BFS tree with constant delay, or take part in competitions, using the multiplexer. Transmissions with constant delay contribute a summand $O(d')$ to the time complexity. If at any time the input value competes with some other l messages, it is promoted to the next level of the tree in time $O(l \log^3 n)$, in view of Lemma 4. Note that two messages competing once will never compete against each other again, since later on, they travel in the same combined message. This means that the total time spent by an input value on competing with other messages is bounded by $O(k \log^3 n)$. Thus the last message arrives at the meeting point in time $O(d' + k \log^3 n)$. This shows that Step 2 is executed in time $O(d' + k \log^3 n)$.

In Step 3, the distribution of the compound message is performed with the help of a broadcasting procedure from [21] in time $O(d' + \log^3 n)$. □

The final M2M multicast in the graph G works in $O(\log n)$ stages. In stage i, Algorithm GRAPH-MULTICAST(C) is performed simultaneously and independently for all clusters C of color i: first from 2-partition $\pi(0)$ and then from

2-partition $\pi(2d)$. (Recall that clusters of the same color and in the same 2-partition are at distance at least 2, hence transmissions in such clusters do not interfere).

Theorem 2. *For any n-node graph G with k participants at distance at most d, M2M multicast can be done in time* $O(d \log^2 n + k \log^4 n)$.

Proof. In view of Lemma 6, there exists a cluster containing all participants. The execution of Algorithm GRAPH-MULTICAST(C) for this cluster completes M2M multicast in the entire graph. It remains to estimate the time complexity of the algorithm.

First note that partitioning of the graph into clusters and cluster coloring does not involve communication and can be done in each node locally because nodes know the topology of the graph. The algorithm works in $O(\log n)$ stages, where the time complexity of each stage is equal to the execution time of the Algorithm GRAPH-MULTICAST(C), for any cluster. The latter is $O(d' + k \log^3 n)$ (by Lemma 7), where $d' = O(d \log n)$ (by Lemma 5). Hence the time complexity of our M2M multicast algorithm is $O(d \log^2 n + k \log^4 n)$. □

4. Conclusion

In this paper we gave an $O(d \log^2 n + k \log^4 n)$-time algorithm for solving the M2M multicast problem for a group of k participants with maximum distance d, in an arbitrary radio network consisting of n nodes. Our approach uses a clustering technique for partitioning the radio network, and a new algorithm for promoting messages in clusters. Interesting problems left for further investigation include (1) tightening the upper bounds on M2M multicast time, (2) establishing lower bounds for M2M multicast time, (3) developing locality-sensitive multicast algorithms for the case when the nodes of the network have only limited (e.g., local) knowledge of the topology, (4) investigating how efficient updating affects performance of multicast in mobile radio networks, and (5) studying randomized algorithms for the M2M multicast problem.

References

[1] B. Awerbuch, D. Peleg, Routing with polynomial communication-space trade-off, SIAM J. Discrete Math. 5 (1992) 151–162.
[4] D. Bruschi, M. Del Pinto, Lower bounds for the broadcast problem in mobile radio networks, Distributed Comput. 10 (1997) 129–135.
[5] I. Chlamtac, S. Kutten, On broadcasting in radio networks-problem analysis and protocol design, IEEE Trans. Commun. 33 (1985) 1240–1246.
[6] I. Chlamtac, O. Weinstein, The wave expansion approach to broadcasting in multihop radio networks, IEEE Trans. Commun. 39 (1991) 426–433.
[7] B. Chlebus, L. Gąsieniec, A. Gibbons, A. Pelc, W. Rytter, Deterministic broadcasting in unknown radio networks, Distributed Comput. 15 (2002) 27–38.
[8] B. Chlebus, L. Gąsieniec, A. Ostlin, M. Robson, Deterministic Radio Broadcasting, in: Proc. 27th Internat. Colloq. on Automata, Languages and Programming, ICALP'00, pp. 717–728.
[9] M. Christersson, L. Gąsieniec, A. Lingas, Gossiping with bounded size messages in ad-hoc radio networks, in: Proc. 29th Internat. Colloq. Automata, Languages and Programming, ICALP'02, pp. 377–389.
[10] M. Chrobak, L. Gąsieniec, W. Rytter, Fast Broadcasting and Gossiping in radio networks, J. Algorithms 43 (2) (2002) 177–189.
[11] A.E.F. Clementi, A. Monti, R. Silvestri, Distributed broadcast in radio networks of unknown topology, Theoret. Comput. Sci. 302 (2003) 337–364.
[12] A. Czumaj, W. Rytter, Broadcasting algorithms in radio networks with unknown topology, in: Proc. 44th Annu. Symp. on Foundations of Computer Science, FOCS'03, pp. 492–501
[13] G. DeMarco, A. Pelc, Faster broadcasting in unknown radio networks, Inform. Process. Lett. 79 (2001) 53–56.
[14] A. DeBonis, L. Gąsieniec, U. Vaccaro, Optimal two-stage algorithms for group testing problems, SIAM J. Comput. 34 (5) (2005) 1253–1270.
[15] D.Z. Du, F.K. Hwang, Combinatorial Group Testing and its Applications, World Scientific, Singapore, 2000.
[16] M. Elkin, G. Kortsarz, Improved broadcast schedule for radio networks, in: Proc. 16th Annu. ACM-SIAM Symp. on Discrete Algorithms, SODA'05, pp. 222–231.
[17] P. Erdös, P. Frankl, Z. Füredi, Family of finite sets in which no set is covered by the union of r others, Israel J. Math. 51 (1985) 75–89.
[18] Z. Füredi, On r-cover free families, J. Combinat. Theory 73 (1996) 172–173.
[19] I. Gaber, Y. Mansour, Broadcast in radio networks, J. Algorithms 46 (1) (2003) 1–20.
[20] L. Gąsieniec, A. Lingas, On adaptive deterministic gossiping in ad hoc radio networks, Inform. Process. Lett. 2 (83) (2002) 89–94.
[21] L. Gąsieniec, D. Peleg, Q. Xin, Faster communication in known topology radio networks, in: Proc. 24th Annu. ACM Symp. on Principles of Distributed Computing, PODC'05, pp. 129–137.
[22] L. Gąsieniec, I. Potapov, Gossiping with unit messages in known radio networks, in: Proc. Second IFIP Internat. Conf. on Theoretical Computer Science, TCS'02, pp. 193–205.

[23] L. Gąsieniec, I. Potapov, Q. Xin, Time efficient gossiping in known radio networks, in: Proc. 11th Colloq. on Struct. Inform. and Comm. Complexity, SIROCCO'04, pp. 173–184, to appear.
[24] L. Gąsieniec, T. Radzik, Q. Xin, Faster deterministic gossiping in ad-hoc radio networks, in: Proc. Ninth Scandinavian Workshop on Algorithm Theory, SWAT'04, pp. 397–407.
[25] W.H. Kautz, R.R. Singleton, Nonrandom superimposed codes, IEEE Trans. Inform. Theory 10 (1964) 363–377.
[26] D. Kowalski, A. Pelc, Faster deterministic broadcasting in ad hoc radio networks, SIAM J. Discrete Math. 18 (2004) 332–346.
[28] D. Kowalski, A. Pelc, Centralized deterministic broadcasting in undirected multi-hop radio networks, in: Proc. Seventh International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX'04, pp. 171–182.
[29] D. Liu, M. Prabhakaran, On randomized broadcasting and gossiping in radio networks, in: Proc. Eighth Annu. Internat. Conf. on Computing and Combinatorics, COCOON'02, pp. 340–349.
[30] A. Sen, M.L. Huson, A new model for scheduling packet radio networks, in: Proc. 15th Annu. Joint Conf. of the IEEE Comp. and Comm. Soc., 1996, pp. 1116–1124.
[32] J.M. Tsai, H.-H. Fang, C.-Y. Lee, A multicast solution for ATM video applications, IEEE Trans. Circuits Systems Video Technol. 7 (1997) 675–686.
[33] Y. Xu, An $O(n^{1.5})$ deterministic gossiping algorithm for radio networks, Algorithmica 36 (1) (2003) 93–96.